Needham, Chapter 5, Section IX, p. 235
G. Palmer, December 16, 2015

Application of Figure [17] to illustrate curvature and circle of curvature. Objects are created for two figures. The second figure is transposed to the right side of Figure 1.

It was more difficult than I anticipated to create this plot, and I am not sure that it is correct. For the curvature on the right hand side of Figure 1, I used the equation

$$\tilde{k}(\hat{\xi}) \equiv \text{Im}\left[\frac{f''\hat{\xi}}{f'|f'|}\right], \quad \text{where } \hat{\xi} = \frac{\vec{\xi}}{|\vec{\xi}|}, \text{ the unit tangent at p, and } \vec{\xi} \text{ and } \hat{\xi} \text{ are complex numbers.}$$

$\tilde{k}(\hat{\xi})$, a scalar, is just the curvature added by the mapping e^z. The original curvature of the slanted line is zero.

To find the center of the circle, multiplication of a small vector parallel to the tangent at p by f'(p) reveals the amplitwist at f(p):

$$f'(p)\,\vec{\xi} = e^{p}\vec{\xi}$$

Then, the length of the teal colored arrow represents the curvature. The red arrow is plotted in the direction of the center of the circle of curvature from f(p). This direction is arg(f'(p) $\vec{\xi}$) + Pi/2, and is implemented in the expression $i\tilde{k}(\vec{\xi})\tilde{\xi}/|\tilde{\xi}|$.* The radius of the circle of curvature is $1/\tilde{k}(\vec{\xi})$.

*See the "kArrow" definition in the (*CURVATURE*) section of the sourcecode. The $\tilde{\xi}$'s are complex vectors. *Mathematica* resists stacking of overset diacriticals, so I have omitted the vector arrow and have not attempted to use the hat for the unit vector.

In[424]:=
```
Clear["Global`*"]

(* PROGRAM FOR FIGURE 1. CIRCLE OF CURVATURE OF e^z at p ON SLANTED LINE *)

(* CONSTANTS *)

pr = 2; (* plot range *)

θ = Pi / 4; (* arbitrary angle *)

z = Exp[I θ]; (* arbitrary vector for slanted line *)

x = 1.5;  (* offset of slanted line at y intercept *)

p = z - x; (* point on slanted line *)

ξ = .5 z; (* arbitrary small complex vector; blue arrow *)

a = .5; (* arbitrary x value on LHS *)
```

```
b = 2 I; (* arbitrary y value on LHS *)

hdSz = .02; (* size arrowheads *)


(* FUNCTIONS *)

v[z_] := {Re @ z, Im @ z}; (* complex to vector *)

f[z_] := Exp[z];

(* sign given as 1 or -1 *)
arc[q_, r_, θ0_, θ1_, sign_] :=
  {Arrowheads @ .02, Arrow[ v @ Table[r Exp[I θ] + q, {θ, Range[θ0, θ1, sign .01]}]]};

k[z_] := Im[(1 / Abs[Exp[z]]) (ξ / (Abs @ ξ))];

(* The dampened sign function arrow is a bit tedius to get
 right. If someone knows a better way, please let me know. *)

squigglyArrow := Module[{points, squigglyLn, hdPt1, head, squigArrow},

   pts = Table[{x, (.5 / (x^4)) Sin[9 Pi x]}, {x, Range[1, 2, .01]}];

   squigglyLn = Line[pts];

   hdPt1 = {2, (.25 / 2^3) Sin[12 Pi 2]};

   head =
    {Arrowheads @ .03, Arrow[{hdPt1, {hdPt1[[1]] + .02, 0}, {hdPt1[[1]] + .18, 0}}]};

   squigArrow = {squigglyLn, head};

   Translate[squigArrow, {-1, 0}]

  ];

SetAttributes[{v, f}, Listable];


(* FRAMES *)

xAxis1 = Line[v @ {-pr, pr / 2}];

xAxis2 = Line[v @ {-pr, pr / 2}];

yAxis = Line[v @ {-pr I / 3, pr I}];

frame1 = {xAxis1, yAxis};

frame2 = {xAxis2, yAxis};


(* POINTS *)
```

```
ptSz = .02;

points1 = {EdgeForm @ Black, FaceForm @ White,
   Table[Disk[v @ pt, ptSz], {pt, {0, p, a, b, a + b}}]};

points2 = {EdgeForm @ Black, FaceForm @ White, Table[Disk[v @ pt, ptSz],
    {pt, Join[f @{p, a + b}, {0, f @ p + I (1 / k[p]) ξTilde / Abs[ξTilde]}]}]};


(* CURVES *)

(* the slanted line *)

d = .01; (* distance between points *)

slantLnPts = Table[a z - x, {a, Range[-.8, 3, d]}];

slantLn = {Line[v @ slantLnPts]};

fSlantLnPts = f @ slantLnPts;

fSlantCrv = { Line[v @ fSlantLnPts]};


(* the vertical line *)

vertLnPts = Table[a + I y, {y, Range[0, pr, d]}];

vertLn = {Dashed, Line[v @ vertLnPts]};


(* the horizontal line *)

horizLnPts = Table[x + b, {x, Range[0, a, d]}];

horizLn = {Dotted, Line[v @ horizLnPts]};


curves1 = {slantLn, vertLn, horizLn};

curves2 = {fSlantCrv};


(* ARROWS, LINES *)

xiArrow = {Blue, Thick, Arrowheads @ hdSz, Arrow[v @ {p, p + ξ}]};

arcArrow1 = arc[p, Abs @ ξ, 0, Arg @ ξ, 1];

arcBase1 = {Dotted, Gray, Line[v @ {p, p + Abs @ ξ}]};

headPos = -IntegerPart[Length[fSlantLnPts] / 4];

slantCrvArrow = {Arrowheads @ hdSz, Arrow[v @ fSlantLnPts[[1 ;; headPos]]]};
```

```
sArrow = Scale[squigglyArrow, 1 / 2];

sArrow = Translate[sArrow, {1, 0}];

dfArrow = {Arrowheads @ hdSz, Arrow[v @ {f[p] + .9 + .9 I, f[p] + f[p] ξ}]};

arrows1 = {xiArrow, arcArrow1, arcBase1, sArrow};

arrows2 = {slantCrvArrow, dfArrow};


(* CURVATURE *)

r = 1 / k[p]; (* The radius of the circle
 of curvature if the reciprocal of the curvature. *)

(* f'(p)ξ⃗, amplitwist on the infinitesimal
 vector ξ⃗ tangent to the straight curve *)

ξTilde = Exp[p] ξ;

unitξTilde = ξTilde / Abs[ξTilde]; (* unit vector tangent to curve at p *)

ξTildeArrow = {Blue, Arrowheads @ hdSz, Arrow[v @ {f @ p, f @ p + ξTilde}]};
(* amplitwisted ξ⃗ *)

center = f @ p + I r unitξTilde; (* center of circle of curvature *)

(* unit vector orthogonal to curve at f(p) multiplied by curvature *)

kArrow = {RGBColor[0.2, 0.5, 0.5],
   Arrowheads @ hdSz, Arrow[v @ {f @ p, f @ p + I k[p] unitξTilde}]};

kCircle = {Dashed, Gray, Circle[v @ center, r]}; (* circle of curvature *)

curvature2 = {kArrow, kCircle, ξTildeArrow};


(* TEXT *)

default = 12;
small = 8;
large = 14;

data1 = {
   {"0", 0 - .1 - .15 I, default},
   {"p", p - .15, default},
   {"ξ⃗", p + ξ / 2 - .15 + .05 I, small},
   {"a", a - .1 I, default},
   {"b", b - .1, default},
   {"ϕ", p + 1.2 (Abs @ ξ) Exp[I Arg[ξ] / 2], small},
   {"e^z", 1.6 + .3 I, default},
```

```
    {"Figure 1. Circle of curvature K at p under e^z on straight line\nk̃(ξ̂)=Im[(1/|
        e^p|)(ξ̂/|ξ̂|)]: " <>
      ToString[k @ p] <> "\nr=1/|k̃(ξ̂)|", .75 pr - pr I / 1.5, large}
   };

data2 = {
    data1[[1]],
    {"e^p", f @ p + .15 + .05 I, default},
    {"f'(p)ξ̂ at f(p)", f @ p + 1.1 + 1.1 I, default},
    {"ik̃(ξ̂)ξ̃̂/|ξ̃̂|", f @ p + I k[p] unitξTilde + .2 + .2 I, default},
    {"K",  f @ p + I r unitξTilde + I r unitξTilde Exp[I Pi / 3], default}
   };

text = Table[Text[Style[item[[1]], item[[3]]], v @ item[[2]]], {item, #}] & /@
    {data1, data2};


(* GRAPHICS *)

Graphics[
 {
  {frame1, curves1, arrows1, points1, text[[1]]},
  Translate[{frame2, curves2, arrows2, curvature2, points2, text[[2]]}, v @ 4]
 }, Background → White, PlotRange → {{-1.5 pr, 3 pr}, {-pr, 1.5 pr}},
 ImageSize → Large]
```